Universiteit Utrecht

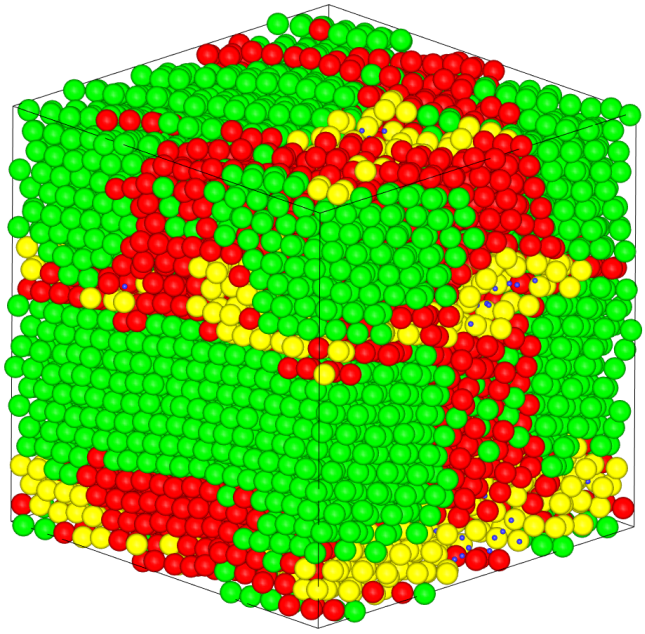
Faculty of Science

Department of Physics

Machine learning reveals new insights into crystal nucleation in a Lennard-Jones fluid

Bachelor Thesis

Bryan Verhoef



Supervisor:

Prof. Dr. Ir. Marjolein Dijkstra

Debye Institute for Nanomaterials Science

June 12, 2020

Abstract

In this thesis we investigate the local crystalline structures on a single-particle level for Lennard-Jones particles during the process of crystal nucleation. We simulate crystal nucleation using Monte Carlo simulations in the NVT ensemble. During the simulations, the local structures around particles are analysed with a Principal Component Analysis and a neural network based classification algorithm. Both analyses show formation of a primarily face-centered cubic and hexagonal close-packed ordered crystal. The Principal Component Analysis suggests that crystal nucleation does not happen via body-centered cubic ordering, although we were unable to properly quantify the degree of body-centered cubic ordering. Lastly we attempt to improve the spatial resolution of the detection of local crystalline structures by using a neural network based autoencoder.

Snapshot of a configuration of particles taken at the end of a simulation. The crystalline ordering of every particle has been classified by a neural network. Green represents face-centered cubic ordering, red represents hexagonal close-packed, yellow represents body-centered cubic and finally particles that are classified as fluid-like have been plotted small and blue. In this research we use a variety of machine learning techniques to identify crystal structures during crystal nucleation of a fluid of Lennard-Jones particles.

Contents

1 Introduction

We are all very familiar with liquid to crystal phase transitions from our experiences from everyday life. But how many of us have actually seen this phase transition happening? For most people, looking at freezing water is probably not the most exciting thing to do. Maybe some have experienced their beverage suddenly and rapidly freezing after taking it out of the freezer. This sudden process is known as crystal nucleation.

A commonly used theoretical model to describe crystal nucleation is Classical Nucleation Theory. At a certain point during cooling of a liquid, the crystal phase becomes the state with a lower free energy. However, when some liquid particles happen to come together and form a small cluster of crystal particles, they also form an interface between the crystal and the liquid phase. This interface is energetically unfavourable and therefore forms a barrier to further growth of that small cluster of crystal particles, also known as a nucleus. The liquid is now left in this metastable state until random fluctuations or external factors, like the introduction of dust particles, manage to push a nucleus over the energy barrier. Once a nucleus has passed this energy barrier, it can continue to grow until the entire liquid has crystallized.

However, the question still remains what type of crystal actually forms during crystal nucleation. Is it also possible that there is an intermediate crystal type on the road to full crystallization?

Alexander and McTague investigated this theoretically in 1978. Using a Landau theory they concluded that the body-centered cubic (bcc) crystal structure is favoured near the melting line. This suggested that crystal nucleation might happen via bcc [1].

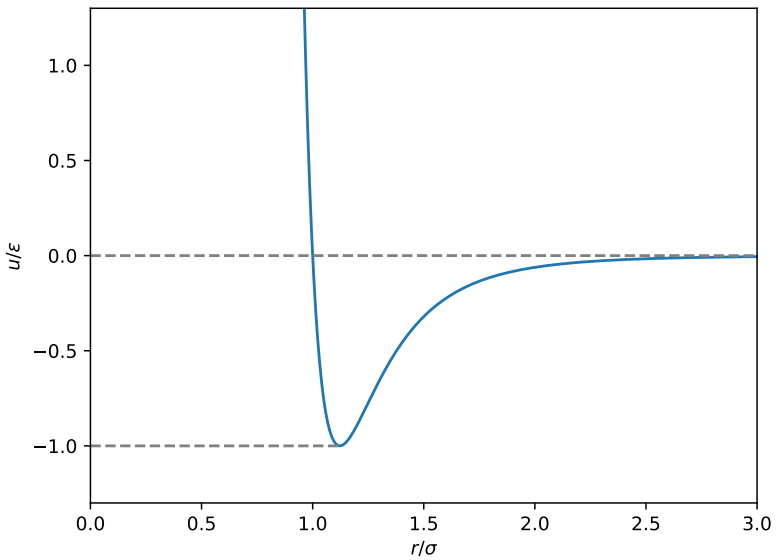


Figure 1: The Lennard-Jones potential $u(r)$ as a function of distance r. The axes are in reduced units.

In this thesis we study by simulations crystal nucleation of Lennard-Jones particles. The Lennard-Jones potential is a very broadly studied interaction potential. As can be seen in figure 1, the Lennard-Jones potential features short-range repulsion and long-range attraction. It is commonly used to model atomic and molecular interactions [2]. The Lennard-Jones potential is also frequently used to test and demonstrate novel simulation techniques [3–5].

Not surprisingly, a lot of simulation work has been done on crystal nucleation of Lennard-Jones particles. Simulations done by Ten Wolde et al. showed that a Lennard-Jones nucleus is primarily structured as face-centered cubic (fcc) in the core whereas bcc ordering was more prominent on the surface of the nucleus [6]. Eslami et al. used a slightly different simulation technique from Ten Wolde et al. when they studied Lennard-Jones crystal nucleation, but they did confirm the earlier conclusions from Ten Wolde et al. about the structure of the Lennard-Jones nucleus [5]. While Ten Wolde et al. simulated 10 648 particles in 1996, Ouyang et al. were able to simulate more than a million particles in 2020. Besides seeing the same structure that Ten Wolde and Eslami found with fcc in the core of the nucleus and bcc on the surface, Ouyang et al. also saw hexagonal close-packed (hcp) structure in the Lennard-Jones nuclei. They also studied particles that they identified as precursors to crystal nucleation and noted that these precursor particles were primarily bcc ordered. From this they conclude that crystal nucleation of Lennard-Jones particles is likely to happen via bcc [7].

A very broadly used set of tools for studying crystal structures are the bond-orientational order parameters [8]. All three previously mentioned simulation studies on crystal nucleation of Lennard-Jones fluids use variations on these bond-orientational order parameters. What these papers also have in common, is that they choose a certain subset of two to four bond-orientational order parameters to study crystal structures. Instead of choosing a small subset of bond-orientational order parameters, we will use Principal Component Analysis to construct a lower dimensional representation of a much larger set of bond-orientational order parameters. We will also use a neural network based classification algorithm to detect crystal structures.

The remainder of this thesis is structured as follows. In section 2 we will first discuss the simulation techniques we used to simulate a nucleation event of a Lennard-Jones fluid. Subsequently, we will elaborate on the bond-orientational order parameters and Principal Component Analysis and give an overview of neural network based machine learning. Also in section 2, we will further discuss how we used the bond-orientational order parameters, Principal Component Analysis and neural networks to identify crystal structures in our simulation. In the subsequent sections we will present our results.

2 Theory and Methods

2.1 Monte Carlo Simulations

In statistical physics, ensemble averages in the NVT ensemble are calculated by

$$\langle A \rangle = \frac{\int d\mathbf{r}^N A(\mathbf{r}^N) \exp\left[-\beta U(\mathbf{r}^N)\right]}{\int d\mathbf{r}^N \exp\left[-\beta U(\mathbf{r}^N)\right]}. \tag{1}$$

Here, $\beta = 1/k_B T$ and $U(\mathbf{r}^N)$ is the total potential energy of the system. It is not feasible to calculate these ensemble averages by conventional numerical integration techniques, since it is a very high dimensional integral.

Instead one might try a straightforward Monte Carlo integration scheme. In such a scheme, one would randomly generate configurations, give them their appropriate Boltzmann factor and calculate the desired ensemble averages along the way. However, particularly at higher densities, there is a relatively high probability of generating a configuration with a low Boltzmann weight. These configurations would only have a small contribution to the ensemble average and a lot of random configurations need to be generated to get an accurate value.

To solve this problem, Metropolis et al. proposed a new algorithm to calculate these ensemble averages in 1953 [9]. Instead of generating random configurations, they proposed choosing configurations with a probability of $\exp[-\beta U(\mathbf{r}^N)]$.

In practice this is done by starting with a certain configuration and moving a random particle a random distance in a random direction. The total potential energy of this new configuration is then calculated. If the new potential energy is lower than the former potential energy, the move is accepted. If the new potential energy is higher than the previous one, the new configuration is accepted with the probability $\exp[-\beta \Delta U]$, where $\Delta U = U(n) - U(o)$ denotes the potential energy difference between the old and new configuration. These conditions for accepting a trial move can be summarized with an acceptance rule [10]:

$$\mathrm{acc}(\mathrm{o} \to \mathrm{n}) = \min(1, \exp[-\beta \Delta U]). \tag{2}$$

Ensemble averages can then be calculated with

$$\langle A \rangle \approx \frac{1}{m} \sum_{i=1}^{m} A(\mathbf{r}_i^N), \tag{3}$$

where m is the total number of trial moves. This ensemble average needs to be updated after every trial move, even after a trial move has been rejected.

The Metropolis algorithm can also be used for Monte Carlo simulations in the NPT ensemble. Besides particle displacement trial moves, the volume of the simulation box also needs to be randomly changed. Volume and particle displacement trial moves are then accepted with the probability given by

$$\mathrm{acc}(\mathrm{o} \to \mathrm{n}) = \min(1, \exp[-\beta \Delta U - P \Delta V + (N+1) \ln(V[\mathrm{n}]/V[\mathrm{o}])]), \tag{4}$$

where V is the volume, P the pressure, $\Delta V = V(n) - V(o)$ the volume difference between the old an new configuration and n and o refer to the new and old configurations respectively [10]. Ensemble averages can then be calculated as in equation (3).

For the simulation box we used periodic boundary conditions to reduce surface effects. To calculate distances between particles we used the nearest image convention. A schematic overview of the nearest image convention can be found in figure 2.

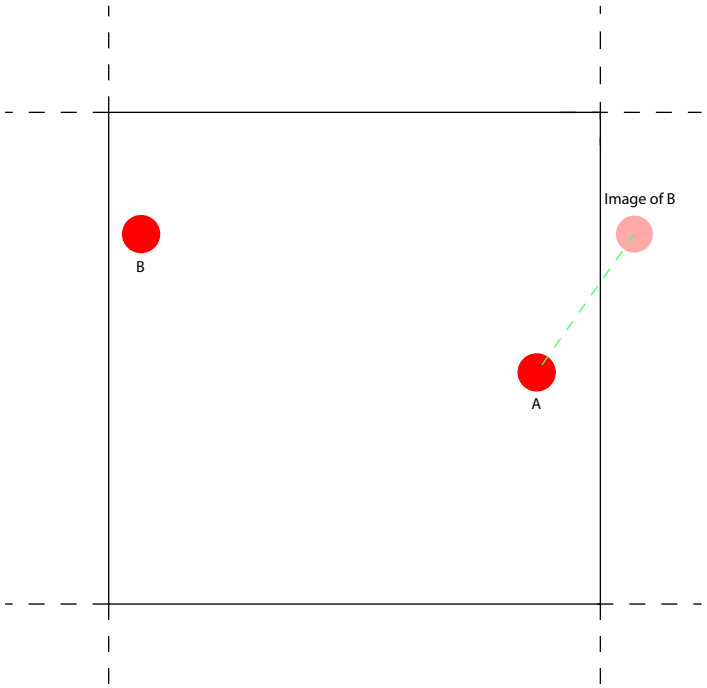


Figure 2: Illustration of the nearest image convention. The solid black lines represent the limits of the simulation volume. If an image of particle B is closer to particle A than particle B itself, the distance between particle A and B is given by the distance between A and the image of B as indicated by the green dashed line.

The potential we used was the truncated and shifted Lennard-Jones potential given by

$$u_{LJ} = \begin{cases} 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] - e_{cut}, & \text{if } r \leq r_{cut} \\ 0, & \text{if } r > r_{cut}, \end{cases} \tag{5}$$

where

$$e_{cut} = 4\epsilon \left[\left(\frac{\sigma}{r_{cut}} \right)^{12} - \left(\frac{\sigma}{r_{cut}} \right)^{6} \right].$$

The e_{cut} term is added such that there is no discontinuity in the potential energy at the cut-off distance. In our simulations we set r_{cut} to 3σ.

The use of a specific cut-off distance for the Lennard-Jones potential in equation (5) allows for an optimisation using cell lists [10]. We consider a simulation consisting of N particles.

After a trial particle displacement $N-1$ Lennard-Jones pair potentials must be recalculated to determine the energy change caused by the trial move. However, many particles will be further than r_{cut} away from the particle that has been moved and thus all these pair potentials are calculated in vain. For the cell list optimisation, the simulation box is divided into cuboidal cells with edges greater than or equal to r_{cut}. This way particles that potentially contribute to the energy difference due to a particle move are all within the same cell as the displaced particle or in one of the 26 cells immediately surrounding the cell containing the displaced particle. Any particle outside any of these cells does not need to be considered at all. The reduction in the number of calculations needed is entirely dependent on the size of the simulation box. If the simulation box is only large enough to fit $3 \times 3 \times 3$ cells, there is no reduction in the calculations of pair potentials possible. As soon as the simulation box can contain more than $3 \times 3 \times 3$ cells, not all pair potentials need to be calculated after a trial particle displacement.

2.1.1 Simulation Details

Configurations of particles used for simulations to study crystal nucleation were initialized as a fluid by randomly placing every particle in the simulation volume. During initialization, two particles can be placed close enough to each other to cause significant overlap and to yield a high potential energy. Because of this the total potential energy was often very high immediately after initialization. If we started measuring ensemble averages right after initialization, we would get a disproportionately large contribution to the ensemble averages from an area of phase space that is very rarely explored during normal Monte Carlo simulations due to its small Boltzmann factor. Therefore it was important to first equilibrate the configuration before starting the measurements of ensemble averages. The exact number of equilibration Monte Carlo trial moves we executed before starting measurements depended on the number of particles in the simulation.

The simulations for studying crystal nucleation were all run in the NPT ensemble. We used 8000 particles in each simulation and equilibrated the system for 12 million Monte Carlo trial moves. We ran four simulations at a pressure of $p\sigma^3/\epsilon = 5.68$ and temperatures of $k_BT/\epsilon = 0.92$, 0.805, 0.69 and 0.575 corresponding to 20%, 30%, 40% and 50% supercooling respectively. These choices for state points were based on the state points used by Ten Wolde et al. [6]. We also used the liquid-solid coexistence line for Lennard-Jones particles determined by Van der Hoef to determine degrees of supercooling [11]:

$$P_{coex} = \beta_m^{-5/4} \exp(-0.4759\beta_m^{1/2})[16.89 + A\beta_m + B\beta_m^2]. \quad (6)$$

In this equation, $\beta_m = 1/k_BT_m$ where T_m is the desired melting temperature, P_{coex} is the corresponding coexistence pressure, $A = -7.2866$ and $B = -2.9895$.

2.2 Measuring Crystal Structure

One property of crystal phases is that there is a degree of local orientational symmetry of the bonds connecting neighbouring particles. In 1983, Steinhardt et al. proposed a set of bond-orientational order parameters to quantify this local orientational symmetry [8]. Subsequently, we can use these bond-orientational order parameters to identify crystal structures

on a per particle basis. Before we can calculate bond-orientational order parameters to quantify local orientational symmetry, a local environment of each particle needs to be defined. One way of doing that is by identifying a set of nearest neighbours for every particle. Unfortunately, there is no unique way of defining a nearest neighbour. Therefore, we will first discuss some ways of defining nearest neighbours before elaborating on the bond-orientational order parameters.

2.2.1 Nearest Neighbours

One straightforward way of defining a nearest neighbour to a particle of interest is by using a simple cut-off radius. All particles that are within that cut-off radius to the particle of interest are then considered nearest neighbours. Benefits of this method are that it is easy to implement and computationally fast to calculate. However, the cut-off radius is an arbitrary tunable parameter. One can use the first minimum in the pair correlation function to set the cut-off radius, but that requires knowledge of the pair correlation function prior to the simulation. In practice this means that we would need to run the simulation first do determine the pair correlation function. The simulation then needs to be repeated to study crystalline structures during the course of the simulation.

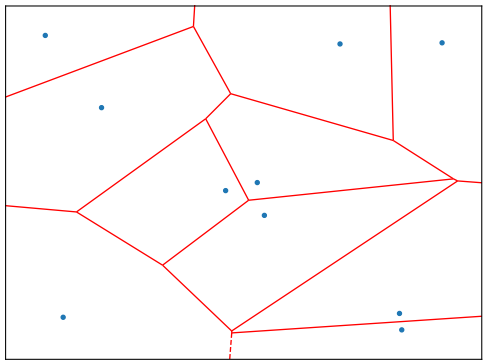


Figure 3: Two dimensional example of Voronoi tessellation. The blue points represent particles and the red lines represent the walls of the Voronoi cells.

Another commonly used definition of nearest neighbours is by means of Voronoi tessellation. To find the nearest neighbours, a Voronoi cell is constructed around every particle. This Voronoi cell consists of all points closer to the particle of interest than to any other particle. The walls of the Voronoi cells are given by the points that are equidistant to the particle of interest and all the particles closest to it. An example of such a construction can be found in figure 3. Two particles are considered nearest neighbours if their Voronoi cells share a wall. Unlike the cut-off radius method, the Voronoi tessellation method does not require a tunable parameter, but it is harder to implement and computationally expensive [12]. With the problems of tunable parameters and computational difficulty in mind, we decided to use the solid angle based nearest neighbour definition proposed by Van Meel et al. in 2012 [12].

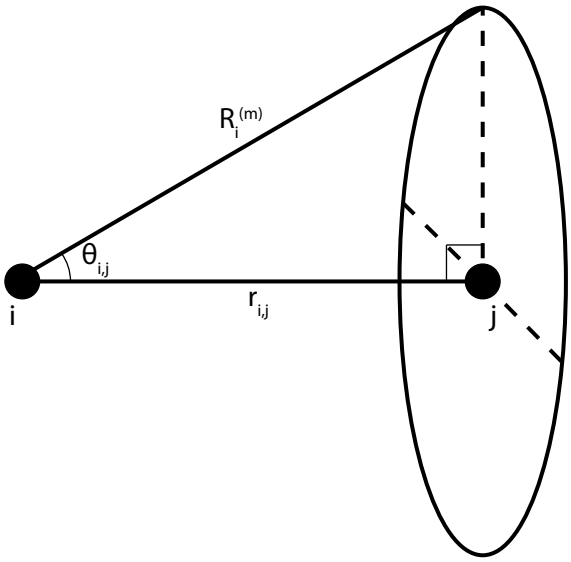


Figure 4: Definition of angle $\theta_{i,j}$ associated with neighbour j. Here $r_{i,j}$ is the distance between particles i and j and $R_i^{(m)}$ is the neighbour cut-off radius of particle i defined by the solid angle nearest neighbour algorithm.

The solid angle nearest neighbour algorithm uses a geometric construction based on solid angles to define a neighbour cut-off radius $R_i^{(m)}$ for each individual particle i. For the algorithm to work, all potential neighbours j of particle i need to be sorted in ascending order in their distances $r_{i,j}$ from the particle of interest. The particle farthest away from particle i that will end up still being within the cut-off radius $R_i^{(m)}$ is labelled particle m. This leads to a relation between m and $R_i^{(m)}$ [12]

$$r_{i,m} \leq R_i^{(m)} < r_{i,m+1}. \tag{7}$$

Now an angle $\theta_{i,j}$ is associated with every potential neighbour. This angle is related to $r_{i,j}$ and the yet to be determined $R_i^{(m)}$ as depicted in figure 4. The solid angle algorithm then defines the neighbourhood of particle i to consist of the closest m particles such that the sum of the solid angles associated with the $\theta_{i,j}$ angles equals 4π [12]

$$4\pi = \sum_{j=1}^{m}(1 - \cos(\theta_{i,j})) = \sum_{j=1}^{m}\left(1 - \frac{r_{i,j}}{R_i^{(m)}}\right). \tag{8}$$

Equations (7) and (8) then lead to

$$R_i^{(m)} = \frac{\sum_{j=1}^{m} r_{i,j}}{m-2} < r_{i,m+1}. \tag{9}$$

Since the potential neighbours j are ordered in ascending distance from particle i, the algorithm can now calculate $R_i^{(m)}$ by iterating over all potential neighbours $j = 1, ..., m$ until equation (9) is satisfied [12].

One potential problem of the solid angle algorithm is that situations can occur where particle A is a neighbour of particle B, but particle B is not a neighbour of particle A. This

problem can be resolved either by removing particle A from the set of neighbours of particle B or by adding particle B to the set of neighbours of particle A. In this thesis we resolve this problem by removing particle A from the set of neighbours of particle B.

2.2.2 Bond-Orientational Order Parameters

Now that we have defined a local environment of each particle in terms of nearest neighbours, we can use the Steinhardt bond-orientational order parameters to quantify the local orientational symmetry [8]. The idea of the bond-orientational order parameters is to expand the local density around a particle, defined by its nearest neighbours, in terms of spherical harmonics. The Steinhardt bond-orientational order parameters are given by

$$q_{lm}(i) = \frac{1}{N_b(i)} \sum_{j=1}^{N_b(i)} Y_{lm}(\boldsymbol{r}_{ij}), \tag{10}$$

where Y_{lm} are the spherical harmonics with l and $m \in [-l, l]$ both integers, $\boldsymbol{r}_{ij}$ the distance vector from particle i to particle j and $N_b(i)$ the set of nearest neighbours of particle i. These $q_{lm}(i)$ are not rotationally invariant, so it matters from what angle a system of particles is considered. However, one can construct a rotationally invariant bond order parameter from q_{lm}:

$$q_l(i) = \sqrt{\frac{4\pi}{2l+1} \sum_{m=-l}^{l} |q_{lm}(i)|^2}. \tag{11}$$

Based on the Steinhardt bond-orientational order parameter, Lechner and Dellago introduced an averaged bond order parameter in 2008 [13]. The aim of this averaged bond order parameter was to improve the accuracy of the crystal structure identification by considering more of the local environment of a particle. The averaged bond order parameters are defined as

$$\bar{q}_{lm}(i) = \frac{1}{N_b(i)} \sum_{j=0}^{N_b(i)} q_{lm}(j). \tag{12}$$

Here $j = 0$ corresponds to particle i itself. A rotationally invariant averaged bond order parameter $\bar{q}_l$ can then be constructed from $\bar{q}_{lm}$ in the same way as in equation (11).

Whereas the non-averaged bond order parameters q_l only consider the first shell of particles around a particle of interest, the averaged bond order parameters $\bar{q}_l$ consider the first two particle shells around the particle of interest. In this way, some spatial resolution is sacrificed in favour of an improved accuracy of crystal structure determination. With q_l and $\bar{q}_l$ we can now actually analyse the local crystalline environment around particles.

2.3 Analysing Local Crystal Structure

First of all, it would be useful to be able to detect whether a particle is more solid-like regardless of the possible crystal structure or looks more like a part of a fluid. Knowledge about this is useful for tracking the crystal nucleation process during the simulation and can be a useful aid in interpreting configurations of particles. Secondly we want to be able to actually determine what type of crystal forms during crystal nucleation. To do that, we apply a Principal Component Analysis and neural network based machine learning algorithms.

2.3.1 Detecting Solid-Like Particles

In 1996, Ten Wolde et al. proposed a method to distinguish solid-like particles from fluid-like particles by measuring the correlation between the structures around two particles [6]. For this purpose, they defined a normalized 13-dimensional complex vector for each particle as

$$d_{6m}(i) = \frac{q_{6m}(i)}{\sqrt{\sum_{m=-6}^{6} |q_{6m}|^2}}. \tag{13}$$

Then they defined a scalar product between two neighbouring particles i and j as

$$S_{ij} = \sum_{m=-6}^{6} d_{6m}(i) \cdot d^*_{6m}(j), \tag{14}$$

where * denotes complex conjugation. Two neighbouring particles i and j can be considered connected if S_{ij} exceeds a certain threshold. In this thesis we set this threshold to 0.7. The probability that two particles are connected is quite high even in the fluid phase. Therefore we need to set a minimum number of particles a particle of interest must be connected to for the particle to be considered solid-like. We set this minimum number of particles to 7. Subsequently we can group solid-like particles into clusters by stating that two solid-like particles are in the same cluster if they are connected to each other.

As an example we applied this definition of solid-like particles and this clustering to a box containing a fluid region and an fcc region. A plot of this configuration can be found in figure 5. All particles in this box are the same, we only plotted the fluid particles smaller to make the solid-like particles more visible. The different colours of the solid-like particles indicate that they belong to different clusters. Notable is the singular particle in the fluid region that has been identified as solid-like. In the fluid there is always a probability that the temporal environment of a particle looks more solid-like than liquid-like due to thermal fluctuations.

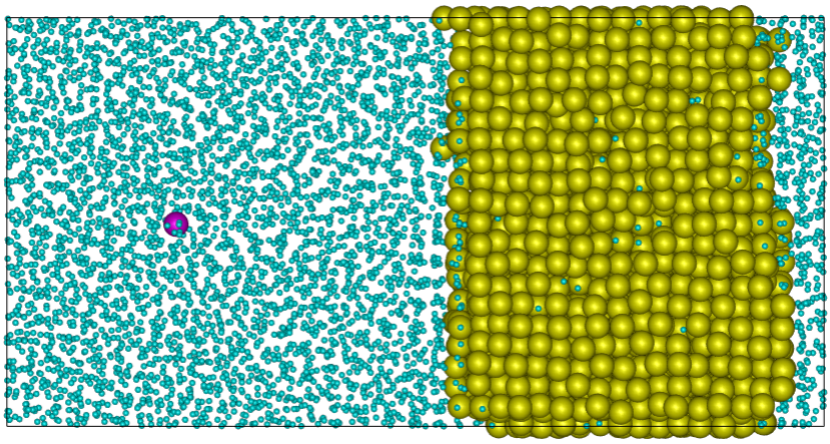


Figure 5: Example of the definition of solid-like particles applied to a configuration containing a fluid and an fcc region. The particles not identified as solid-like are blue and have been plotted smaller to make the solid-like particles more visible. The different colours of solid-like particles denote that these solid-like particles are in different clusters of solid-like particles (two clusters here visible).

2.3.2 Principal Component Analysis

Given a dataset that is described by a set of parameters, the aim of Principal Component Analysis (PCA) is to find linear combinations of parameters that best describe the dataset. A visual illustration of that principle can be found in figure 6. The dataset in figure 6 is represented by two parameters. One can see that most of the variation in the dataset is along the axis indicated by the green arrow. PCA aims to identify this green arrow as the most important axis to describe the dataset, also called the first principal component. All principal components should be orthogonal. In the case of figure 6 the second principal component is indicated by the red arrow.

PCA is especially useful for datasets described by a large set of parameters. The dimensionality of the dataset can then be reduced by projecting the dataset on the first few principal components, discarding the less important principal components. Even though some principal components are discarded, all the original parameters are still represented in the remaining principal components since principal components are linear combinations of the original parameters.

Mathematically PCA is an eigendecomposition problem. Suppose we are studying a system of N particles described by j parameters. The system can then be represented as an $N \times j$ matrix $\boldsymbol{X}$. First we can subtract the mean value for every parameter from the dataset to center the dataset. Then we can calculate the $j \times j$ covariance matrix $\boldsymbol{C_X}$ for our dataset $\boldsymbol{X}$:

$$\boldsymbol{C_X} = \frac{1}{N-1}\boldsymbol{X}\boldsymbol{X^T}.$$

$\boldsymbol{C_X}$ contains information about the correlation between all the pairs of parameters of our system. The diagonal elements represent variances of parameters whereas the off-diagonal elements represent covariance between pairs of parameters. A high covariance between two

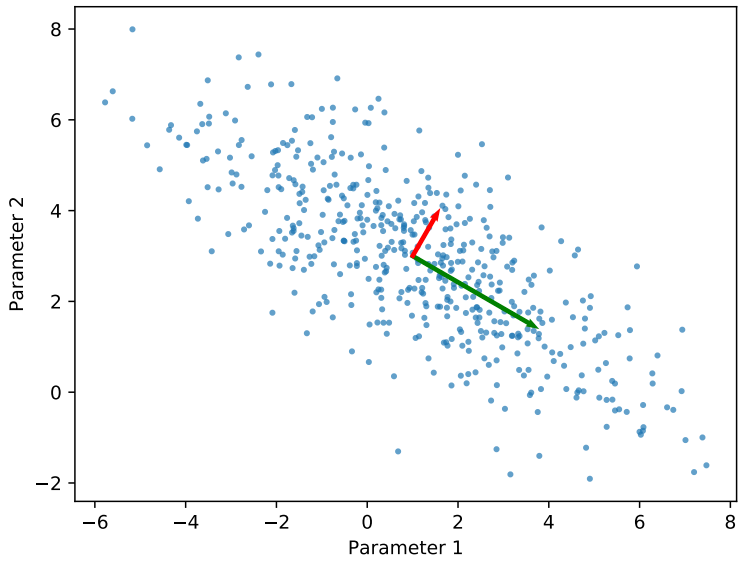


Figure 6: Demonstration of PCA on a two dimensional dataset. Most of the variation of the dataset is along the green axis. Therefore the green axis would be the first principal component [14].

parameters means that the value of one parameter can predict the value of the other parameter. In that sense having both parameters is somewhat redundant in that case. It would be better if we could diagonalize this covariance matrix.

To paraphrase the problem, we need to find a projection matrix $\boldsymbol{P}$ such that $\boldsymbol{Y} = \boldsymbol{PX}$ and $\boldsymbol{C_Y} = \frac{1}{N-1}\boldsymbol{YY^T}$ is diagonalized [15]. We can rewrite $\boldsymbol{C_Y}$ to

$$\boldsymbol{C_Y} = \frac{1}{N-1}\boldsymbol{P}(\boldsymbol{XX^T})\boldsymbol{P^T}.$$

Note that the covariance matrix is a symmetric matrix and symmetric matrices are diagonalized by a matrix of its eigenvectors. Therefore we must be able to write

$$\boldsymbol{XX^T} = \boldsymbol{EDE^{-1}},$$

where $\boldsymbol{E}$ is the matrix with orthonormalized eigenvectors and $\boldsymbol{D}$ the diagonal matrix of eigenvalues. Now we can choose $\boldsymbol{P} = \boldsymbol{E^T}$ and using the fact that the inverse of an orthogonal matrix is just its transpose, we see that

$$\boldsymbol{C_Y} = \frac{1}{N-1}\boldsymbol{D}.$$

From this expression we can conclude that the principal components of $\boldsymbol{X}$ are given by the eigenvectors of $\boldsymbol{XX^T}$ and that the eigenvector corresponding to the largest eigenvalue is the first principal component, since the eigenvalue represents the variance along the corresponding principal component [15].

To determine what crystal types appear during the course of the simulations, we ran a PCA on some known configurations. We performed four separate simulations each containing approximately 4000 particles and we prepared the system in a fluid, fcc, bcc and lastly in an hcp configuration. We equilibrated the four simulation volumes separately in the NPT ensemble at $k_BT/\epsilon = 1.20$. The pressure for the fcc simulation was set to $p\sigma^3/\epsilon = 11.43$, for hcp to $p\sigma^3/\epsilon = 12.77$ and for bcc to $p\sigma^3/\epsilon = 13.83$. This corresponded to 18%, 22% and 25% supercooling respectively according to equation (6). The fluid simulation was run at $p\sigma^3/\epsilon = 2.64$ which leads to the temperature being 35% above the liquid-solid coexistence line. After equilibration, we combined the four volumes and ran two principal component analyses on the entire configuration. One PCA used the non-averaged bond order parameters $q_0, q_1, ..., q_{12}$ and the other used the averaged bond order parameters $\bar{q}_0, \bar{q}_1, ..., \bar{q}_{12}$. After each PCA, we projected the four different configurations onto the space spanned by the resulting first two principal components.

We used the Gaussian Mixture Model implementation from scikit-learn to fit four Gaussian functions to the two-dimensional projections onto PCA space of the four configurations [16]. This Gaussian Mixture Model enables us to determine membership of particles from simulation results to one of the four structures.

2.3.3 Neural Networks

Neural networks are networks of computational units, called neurons, that are inspired by biological networks of neurons. Neural networks can be used for both supervised and unsupervised machine learning. Both methods have been used to identify structures in configurations of particles [17, 18]. In this section we will focus on using neural networks for supervised learning. In supervised learning, a labelled training dataset is required.

The goal of training a neural network with labelled data is to fit a non-linear function to the training data. That non-linear function can then be used to predict the output for previously unseen data. Fully-connected feed-forward neural networks, as the networks used in this thesis, generally consist of three parts. The first part is the input layer containing one or more input neurons. This layer feeds the features of the input data into the network. The input layer has one neuron for each feature of the input data. In this thesis that would be one neuron for each q_l or $\bar{q}_l$. The neural network also has one or more hidden layers that perform the non-linear fitting. Finally, the data is transformed into the desired output format in the output layer. A schematic overview of such a neural network is shown in figure 7.

Consider a fully-connected feed-forward neural network consisting of L hidden layers where layer n contains $I^{(n)}$ neurons. The inputs of each neuron are weighted and added together. Then a bias is added to the sum of weighted inputs. Finally, the output $a_i^{(n)}$ of each neuron is determined by applying an activation function $g^{(n)}(x)$:

$$a_i^{(n)} = g^{(n)}\left(\sum_{j=1}^{I^{(n-1)}} w_{ij}^{(n)} a_j^{(n-1)} + b_i^{(n)}\right). \tag{15}$$

The index n indicates the layer for which the outputs are calculated, the index i refers to each neuron in layer n and j is summed over all the neurons in the preceding layer. The

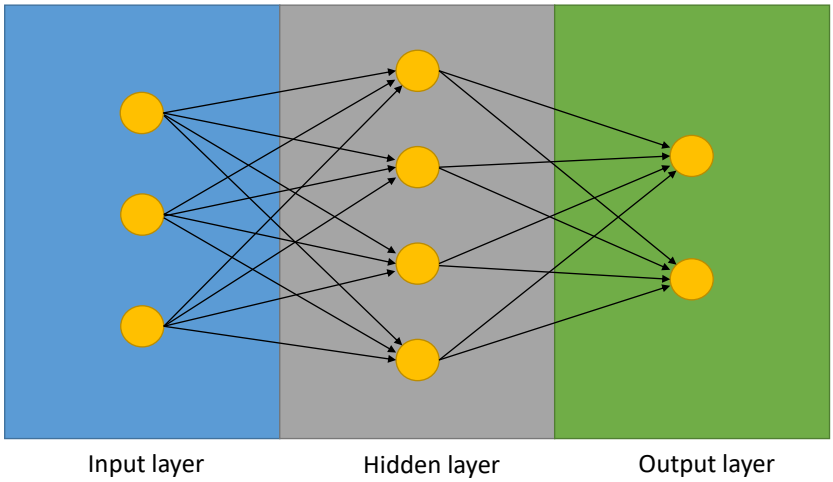


Figure 7: Example of a fully-connected feed-forward neural network with one ($L = 1$) hidden layer.

weights are denoted by w and the biases by b. The non-linearity of the neural network is achieved by using non-linear activation functions $g^{(n)}(x)$.

To train the network we need to determine how well the predicted output of the network matches the labels of the training data. We can then adjust the weights and biases in the network to improve the match. To this end we need to define a cost function C that measures how well the predicted output matches the desired output given by the labels of the training data. The cost function used in this thesis is the mean squared error (MSE) given by

$$\text{MSE} = \frac{1}{2N}\sum_{i=1}^{N}||\mathbf{y}(i) - \mathbf{a}^L(i)||^2, \tag{16}$$

where N is the number of training examples, $\mathbf{a}^L$ is the vector containing the output of the neural network and $\mathbf{y}$ is the desired output of the training data.

The weights and biases can be adjusted to minimize the cost function C with the gradient descent algorithm. This algorithm updates the weights and biases as follows

$$w_{ij}^{(n)} \to w_{ij}^{(n)} - \alpha\frac{\partial C}{\partial w_{ij}^{(n)}}, \tag{17}$$

$$b_{i}^{(n)} \to b_{i}^{(n)} - \alpha\frac{\partial C}{\partial b_{i}^{(n)}}, \tag{18}$$

where α is the learning rate. All training examples are evaluated before the weights and biases are updated. Repeating this procedure should further minimize the cost function. One cycle of evaluating all training examples is commonly referred to as an epoch.

Before training can start, the weights of the network need to be randomly initialized and the biases need to be set to zero. However, random initialization of the weights can cause the gradients of the cost function to become small. The updates to the weights and biases during training will then be very small and training will happen slowly. To overcome this problem, Glorot and Bengio proposed to initialize the weights in layer n by drawing them from a normal distribution with variance $\sigma^2 = 2/(I^{(n-1)} + I^n)$ [19]. This initialization scheme is known as Xavier initialization.

To avoid overfitting of the neural network, we added a weight decay regularization term to the cost function. The cost function is then given by

$$C = \text{MSE} + \frac{\nu}{2} \sum \left(w_{ij}^{(n)} \right)^2, \tag{19}$$

where the sum is over all weights in the neural network and ν is the weight decay parameter.

To improve the gradient descent algorithm's efficiency, we added a momentum term to equation (17). The momentum takes the update to the weights in the previous gradient descent step into account [20]. With the momentum and weight decay terms, equation (17) becomes

$$w_{ij}^{(n)} \to w_{ij}^{(n)} - \alpha \frac{\partial \text{MSE}}{\partial w_{ij}^{(n)}} - \alpha \nu w_{ij}^{(n)} + \eta \Delta w_{ij,prev}^{(n)}, \tag{20}$$

where η is the momentum parameter and $\Delta w_{ij,prev}^{(n)}$ the update to weight $w_{ij}^{(n)}$ during the previous gradient descent iteration.

To reduce the probability that the gradient descent algorithm only finds a local minimum in the cost function, we implemented the mini-batch stochastic gradient descent algorithm. In this version of the gradient descent algorithm, the training data is randomly divided in batches of approximately equal size. After evaluating all the training examples in one batch, the weights and biases are updated. At the end of an epoch all the training examples are split into new batches.

2.3.4 Neural Networks for Analysing Local Crystalline Structures

For the classification of structures in configurations of particles we used a neural network with one hidden layer. The input layer contained 12 neurons for $\bar{q}_l$ with $l = 1, 2, ..., 12$. For the hidden layer we used 120 neurons with a hyperbolic tangent as activation function. The output layer consisted of four neurons, one for fluid, bcc, fcc and hcp. We want to study how well the neighbourhood around a particle matches each of the four structure classes. To this end we used the logistic function given by

$$g(z) = \frac{1}{1 + e^{-z}}$$

as the activation function. This will give a confidence level between 0 and 1 for each of the four structure classes. Note that the output vector is not normalized, so these confidence levels cannot be interpreted as a discrete probability distribution.

The network was trained with the four configurations we used for PCA. The averaged bond order parameters were centered such that the means of all $\bar{q}_l$ over the entire training

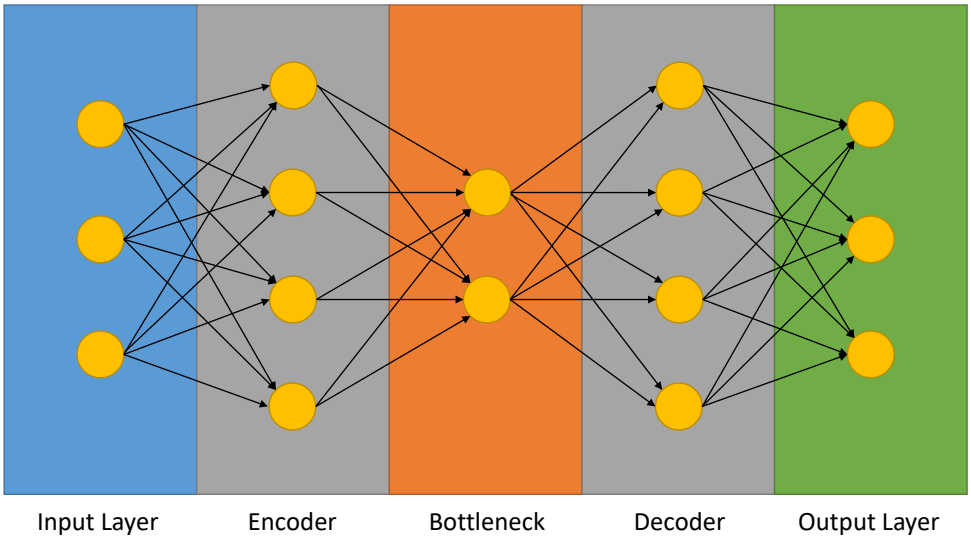


Figure 8: Example of an autoencoder neural network. The network tries to reproduce its input at the output layer. The bottleneck forces the network to learn a lower dimensional representation of the input dataset.

dataset equalled 0. Fluid particles were labelled as $(1, 0, 0, 0)$, fcc particles as $(0, 1, 0, 0)$, bcc particles as $(0, 0, 1, 0)$ and lastly hcp particles as $(0, 0, 0, 1)$. The momentum η was set to 0.5, the weight decay parameter ν to 10^{-4} and the learning rate was $\alpha = 5 \times 10^{-8}$.

To study if the non-linearity of neural networks can improve the projection of the bond order parameters to a two dimensional space compared to the projection obtained from PCA, we constructed an autoencoder neural network. An example of an autoencoder neural network is shown in figure 8.

The autoencoder is an example of unsupervised machine learning. It is trained to reproduce the input at the output layer. Because there is a bottleneck in the architecture that has fewer neurons than the input and output layers, the autoencoder is forced to learn a lower dimensional representation of the input dataset. After training, the encoder part of the autoencoder can be used to project data on this lower dimensional space.

For the autoencoder, we used the same architecture that Boattini et al. used [18]. Our input and output layers have 12 neurons. The encoder and decoder layers both contain 120 neurons and the bottleneck layer contains 2 neurons. The encoder and decoder layers use hyperbolic tangent activation functions while the bottleneck and output layers use linear activation functions. The network was trained with the same dataset we used for PCA. The averaged and non-averaged bond order parameters of the training data were again centered. The learning rate α was set to 5×10^{-7} and the momentum and weight decay settings were the same as the settings used with the neural network for classification.

3 Results and Discussion

In this section, we will first give an overview of the simulations we have run. Then we will present the results obtained from the PCA space projections. Lastly, we will discuss the results from the application of various neural network architectures on identifying crystal structures.

3.1 Simulation Overview

We kept track of the progress of the simulations by recording the number of particles in the largest cluster of solid-like particles in the system. Plots of this parameter during the four simulations can be found in figure 9.

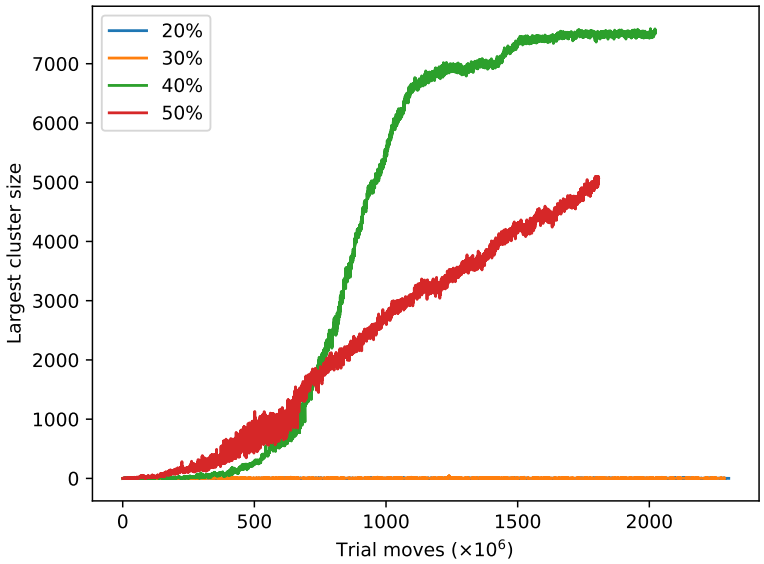
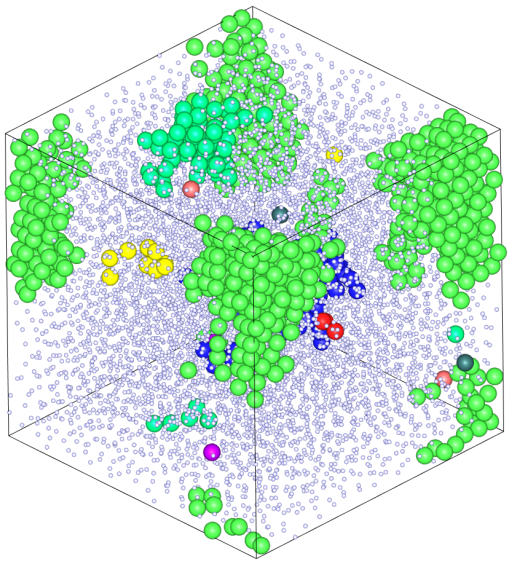


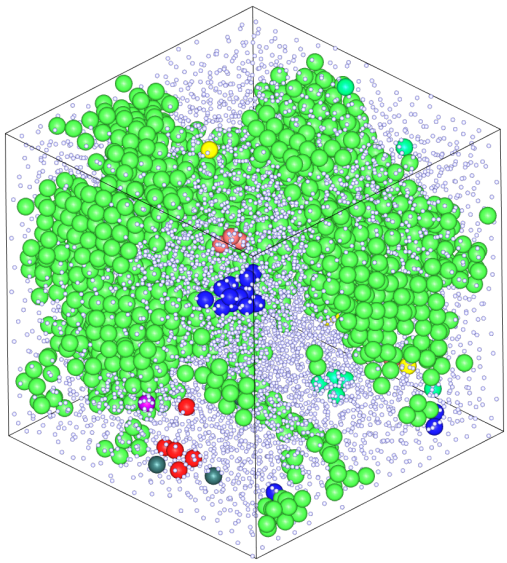
Figure 9: The number of particles in the largest cluster of solid-like particles plotted against the number of trial moves. The simulations at 40% and 50% supercooling crystallized spontaneously while the simulations at 20% and 30% supercooling did not.

Spontaneous crystallization did not occur during the simulations at 20% and 30% supercooling. It seems from the curves for 40% and 50% supercooling in figure 9 that the crystallization of the systems at 40% and 50% supercooling happened via different mechanisms. To get a better understanding of what was happening during these two simulations, we plotted the configuration of particles after 600 million trial moves from both simulations in figure 10. For the configuration from the 50% supercooling simulation in figure 10b, it is notable that the largest cluster of solid-like particles is irregularly shaped. This suggests that solid-like clusters started forming throughout the simulation volume as opposed to one cluster forming and subsequently growing as would be expected in crystal nucleation. Possibly, the state point at 50% supercooling corresponds to an unstable state. In an unstable state, a system will immediately start rearranging itself towards the configuration with the lowest free

energy. In contrast, crystal nucleation is associated with a metastable state. The state point corresponding to 50% supercooling might not be appropriate to study crystal nucleation. The configuration after 600 million trial moves from the simulation at 40% supercooling in figure 10a shows more spherically shaped solid-like clusters while also showing several large clusters. During crystal nucleation we would expect to see only a single large cluster. Therefore it is unclear whether the state corresponding to 40% supercooling is metastable or unstable.



(a) Typical configuration of a Lennard-Jonese fluid from a simulation at 40% supercooling ($k_BT/\epsilon = 0.69$ and $p\sigma^3/\epsilon = 5.68$).



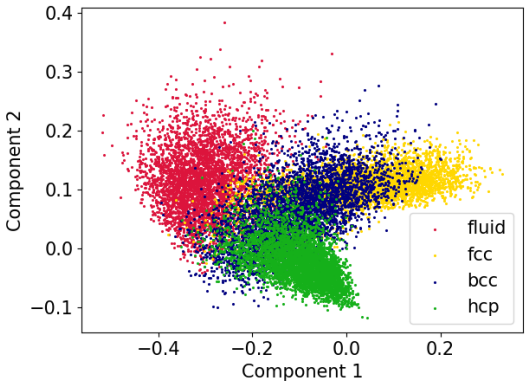
(b) Typical configuration of a Lennard-Jonese fluid from a simulation at 50% supercooling ($k_BT/\epsilon = 0.575$ and $p\sigma^3/\epsilon = 5.68$).

Figure 10: Plots of the configuration of Lennard-Jones particles during the simulations. Both snapshots were taken after 600 million trial moves. The particles that are not identified as solid-like are plotted smaller to make the solid-like clusters more visible. The different colours denote that particles belong to different clusters of solid-like particles.

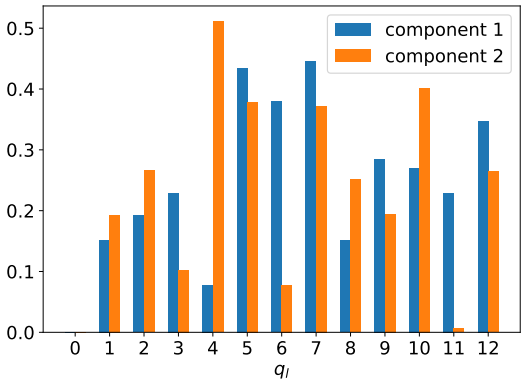
It would be better if we could observe crystallization at 20% or 30% supercooling. However, even after 21 days of running these simulations no crystallization occurred. In future simulations we could study crystal nucleation in a Lennard-Jones fluid by implementing a biasing scheme. This way energetically unfavourable areas of phase space can be explored and the system can be forced to form and subsequently grow a crystal nucleus. For the remainder of this thesis, we will focus on the results from the simulation at 40% supercooling.

3.2 PCA Projections

The first method for identification of local crystal structures we consider is Principal Component Analysis.



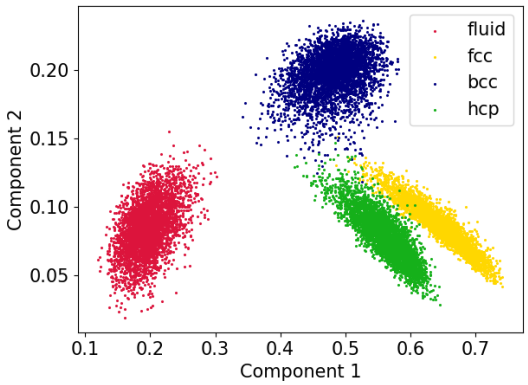
(a) q_l of four different configurations projected onto two-dimensional PCA space.



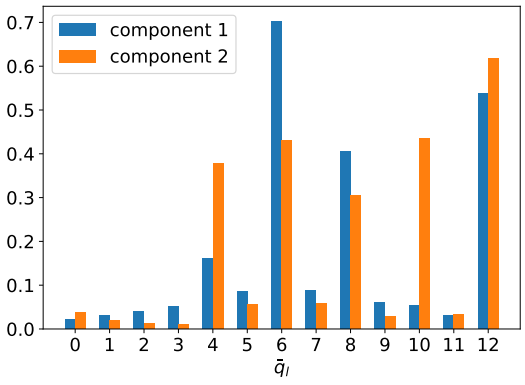
(b) Absolute values of the coefficients of the first and second principal components.

Figure 11: Results of the Principal Component Analysis using the non-averaged bond order parameters. The four different configurations are not well separated.

The results for the PCA using the non-averaged bond order parameters are displayed in figure 11. Notable is that the four different configurations projected onto this PCA space are not well separated. Therefore this analysis is not appropriate for identifying crystal structures in the crystal nucleation simulations. In figure 11b, we plot the coefficients of the first and second principal components and find, from the first principal component, that q_6 and q_{12} are important as expected, but q_5 and q_7 are also important.



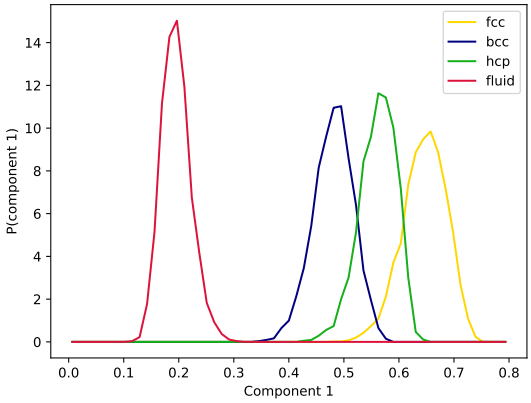
(a) $\bar{q}_l$ of four different configurations projected onto two-dimensional PCA space.



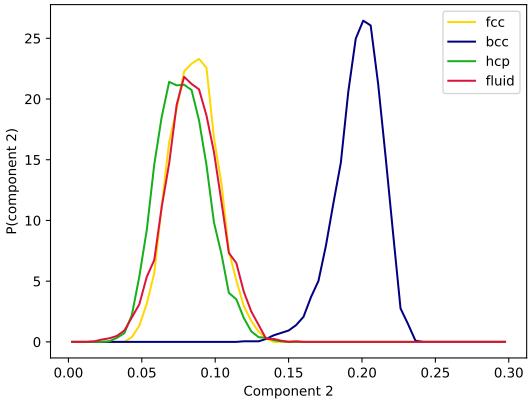
(b) Absolute values of the coefficients of the first and second principal components.

Figure 12: Results of the Principal Component Analysis using the averaged bond order parameters. The four different configurations are reasonably well separated.

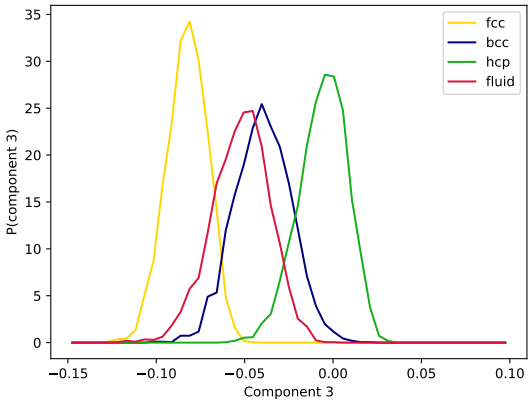
As can be seen in figure 12, when the averaged bond order parameters are used for the PCA, the four configurations are well separated on the resulting PCA space. Therefore, this PCA plane is more suitable for identifying crystal structures during crystal nucleation than the PCA plane obtained with the non-averaged bond order parameters. Figure 12b shows that $\bar{q}_6$ and $\bar{q}_{12}$ are important. In contrast to the principal components obtained from the non-averaged bond order parameters, none of the odd averaged bond order parameters are important. To gain more insight into the principal components, we calculated the probability distributions for the first four principal components. These are shown in figure 13.



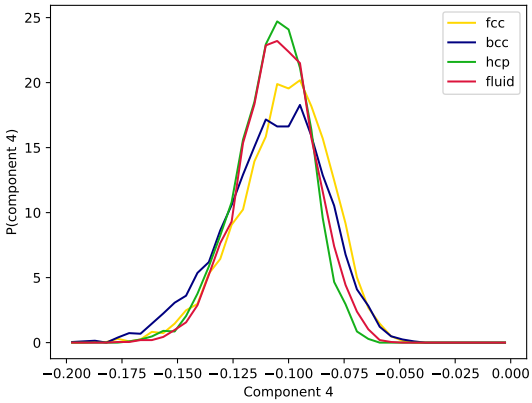
(a) Probability distribution functions of the first principal component.



(b) Probability distribution functions of the second principal component.



(c) Probability distribution functions of the third principal component.

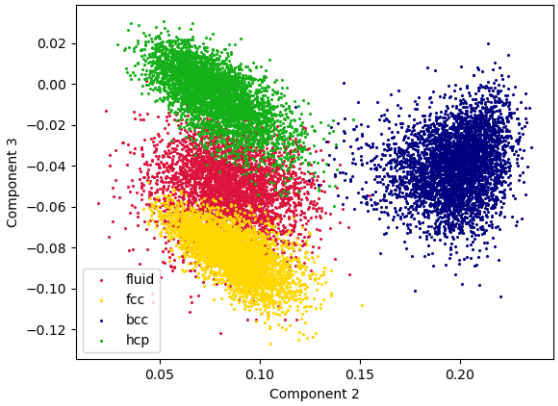


(d) Probability distribution functions of the fourth principal component.

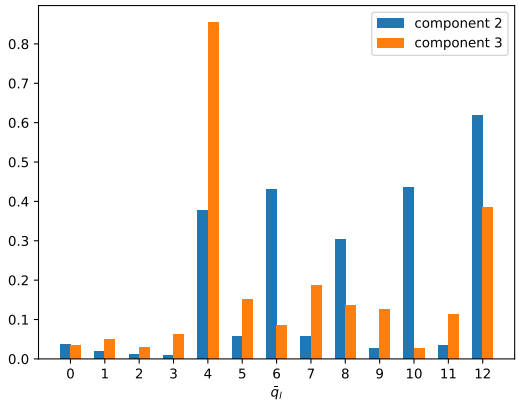
Figure 13: Probability distributions of the first four principal components from the Principal Component Analysis of the averaged bond order parameters $\bar{q}_0, \bar{q}_1, ..., \bar{q}_{12}$. A configuration containing fcc, bcc, hcp and fluid regions was analysed.

From the probability distribution for the first principal component in figure 13a we can see that the first principal component is useful in distinguishing the fluid from the crystal phases while figure 13b shows that the second component can be used to distinguish bcc

structures from fcc and hcp structures. Figure 13c shows that the third component may be useful for distinguishing fcc from hcp structures. In figure 13d we can see that the fourth component is not able to distinguish any of the structures. These probability distributions suggest that a two-dimensional space spanned by the second and third principal components may be useful for distinguishing the three crystal structures. This PCA space is shown in figure 14.



(a) $\bar{q}_l$ of four different configurations projected onto two-dimensional PCA space.



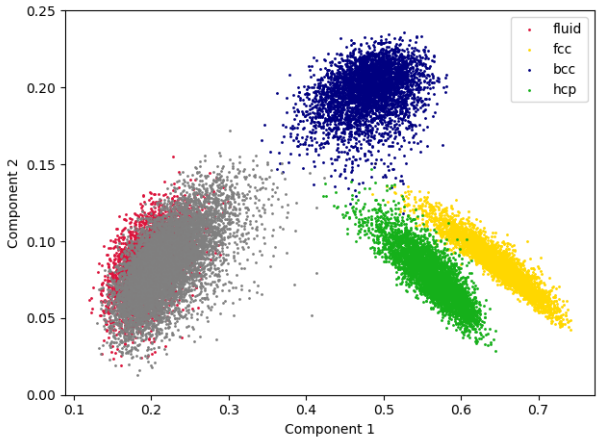
(b) Absolute values of the coefficients of the second and third principal components.

Figure 14: Results of the Principal Component Analysis using the averaged bond order parameters. The three crystal regions are well separated from each other. The fluid region has overlap with the fcc and hcp regions.

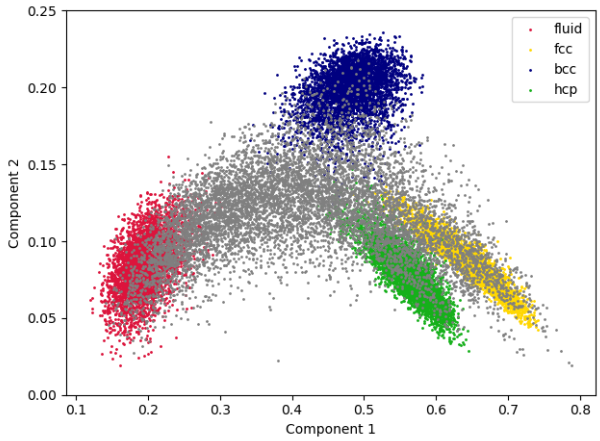
The projection of the four different configurations onto the two-dimensional space spanned by the second and third principal components in figure 14a shows that the three crystal regions in this space are well separated. However, the fcc and fluid regions overlap as well as the hcp and fluid regions. Therefore, this space is less suitable for studying configurations containing fluid-like particles. Since we are simulating crystal nucleation of a Lennard-Jones fluid, we will project simulation results onto the PCA space spanned by the first and second principal components.

Over the course of the simulation at 40% supercooling, we took 840 snapshots of the configuration of Lennard-Jones particles at intervals of 2 million trial moves. Using the averaged bond-orientational order parameters, we projected every snapshot onto the two-dimensional PCA space in figure 12a.

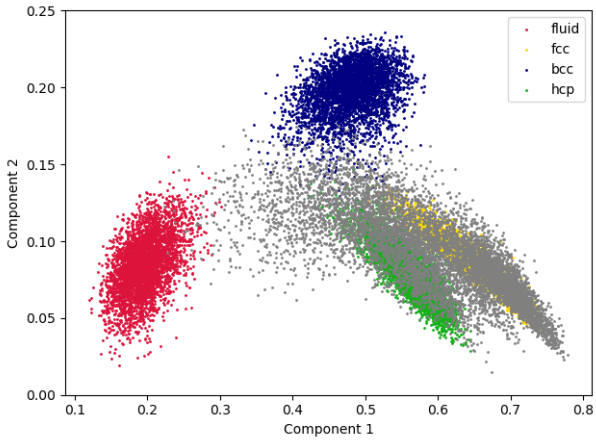
In figure 15 we plotted the projections of three snapshots onto the PCA space. From these projections it seems that the crystals formed at the end of the simulation consist of hcp and fcc ordered particles. The presence of hcp crystal structures could be caused by the system being unstable at 40% supercooling. The main difference between hcp and fcc crystal structures is the stacking of crystal planes. In hcp crystals, the crystal planes follow an ABAB stacking pattern, while fcc crystal planes are ABCABC stacked. In an unstable system, crystal structures form independently throughout the system. When initially separate structures meet, they might not match up to form a contiguous fcc structure and locally



(a) Snapshot after 20×10^6 trial moves.



(b) Snapshot after 900×10^6 trial moves.



(c) Snapshot after 1680×10^6 trial moves

Figure 15: Projections of averaged bond order parameters from the simulation at 40% supercooling onto a two-dimensional PCA space. The PCA space was obtained by running a Principal Component Analysis on a configuration containing fluid, fcc, bcc and hcp particles.

form an hcp structure or vice versa. However, Ouyang et al. also reported hcp structures in their crystal nuclei while their molecular dynamics simulations were run at a state point corresponding to 20% supercooling [7]. Therefore, it is still possible that our simulation was run at a metastable state point.

From figure 15 it also seems that most particles do not pass through the bcc region of the PCA space. To further quantify this, we applied a Gaussian Mixture Model to fit two-dimensional Gaussian functions to the four regions of the PCA space. Then we use the Gaussian Mixture Model to classify each particle in a snapshot as fluid, bcc, hcp or fcc.

The results of this classification process for one snapshot are shown in figure 16. The Gaussian Mixture Model classifies every particle as bcc with probabilities close to 1 at least at some point during the simulation. However, this does not necessarily mean that crystal nucleation of Lennard-Jones particles happens via bcc since the Gaussian Mixture Model determines the discrete membership probability distributions for each particle over the course

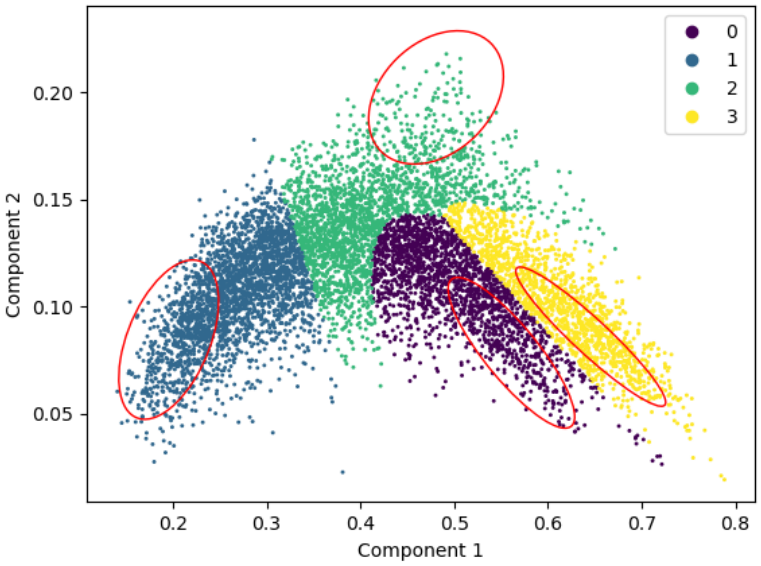


Figure 16: Classification of the particles from the snapshot after 900×10^6 trial moves by the Gaussian Mixture Model. The red lines indicate the 2σ ellipses of the Gaussians fitted to the four regions in the PCA space. Gaussian Mixture component 0 corresponds to hcp, 1 to fluid, 2 to bcc and 3 to fcc.

of the simulation. If, for example, the probabilities of a particle belonging to either the fluid, hcp or fcc clusters is near zero, the probability of the particle to be associated with the bcc cluster must be near one. Because of this normalization, even particles that are far away from the bcc cluster can still have a high probability of being associated to the bcc cluster. Therefore, this method is not appropriate to determine whether crystal nucleation of Lennard-Jones particles happens via bcc.

One way to still use the Gaussian Mixture Model to quantify whether crystal nucleation happens via bcc is to evaluate the Gaussian function belonging to the bcc cluster for every particle. This value can then serve as a measure of how similar to bcc the particle is. Interpretation of this measure is not straightforward and requires further consideration.

3.3 Neural Network Classification

Unlike the results of the Gaussian Mixture Model, the output layer of the neural network we use for classification is not normalized to represent a probability distribution. Possibly, this will allow us to determine to what extent the structure around a particle corresponds to one of the three crystal structures or to a fluid.

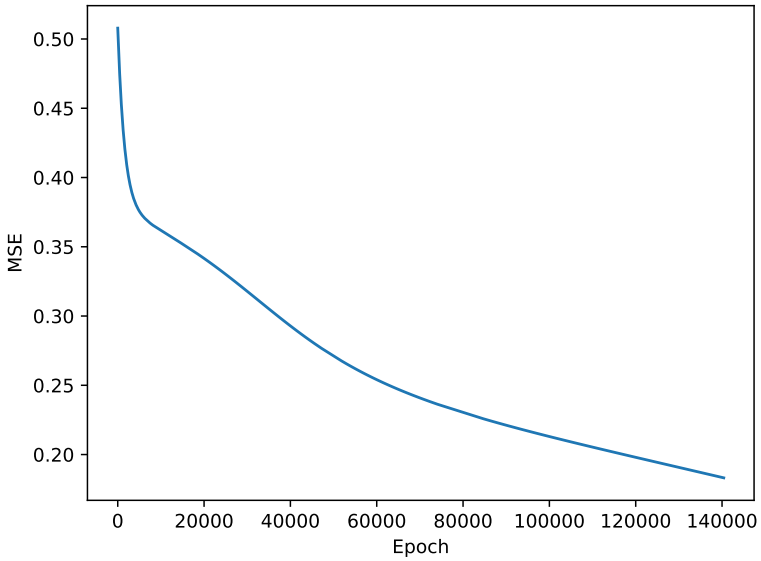


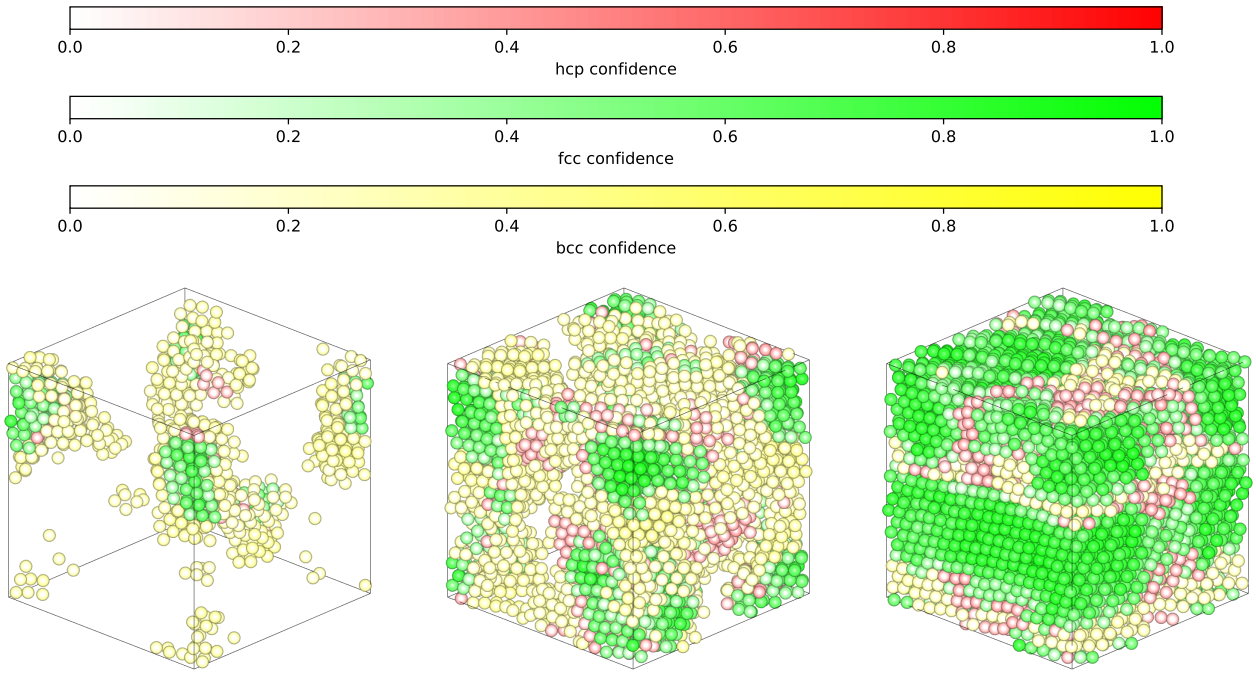
Figure 17: Plot of the Mean Squared Error of the neural network for classification over the course of training the network.

The MSE of the neural network during training is shown in figure 17. It seems that the minimum of the MSE has not been reached. We might be able to improve classification accuracy of the neural network by training it longer.

To visualize the results of the classification by the neural network in a plot of a configuration, we assign to each class a different base colour. Particles that have the highest confidence of being part of an fcc structure are green. Red represents hcp and yellow represents bcc. For clarity, all particles that are classified as fluid-like particles are omitted from these configuration plots. The level of confidence is subsequently linearly represented by the saturation of the colour. If the confidence is high, the colour will appear more saturated whereas the colour will appear more white if the confidence is low.

Figure 18 shows the results of classification of particles from three snapshots taken during the simulation at 40% supercooling. Notable is that the centres of the nuclei are classified with the highest confidence. These plots also suggest that the centres of the nuclei are mostly fcc ordered. This is in agreement with Ten Wolde, Eslami and Ouyang [5–7]. The surfaces of the crystal nuclei are classified as bcc ordered, although with lower confidence.

The classification by the neural network is plotted on the same PCA space as the classification by the Gaussian Mixture Model in figure 19 to make comparison between the two easier.



(a) Snapshot at 6×10^6 trial moves.

(b) Snapshot at 900×10^6 trial moves.

(c) Snapshot at 1680×10^6 trial moves.

Figure 18: Classification by neural network with logistic activation function at the output layer. Green denotes fcc, red denotes hcp and yellow denotes bcc. Particles classified as fluid-like are omitted. The saturation of the colours linearly corresponds to the confidence of classification, where low saturation indicates low confidence. Snapshots are taken during simulation at 40% supercooling.

The classification by the neural network shows some notable differences to the classification by the Gaussian Mixture Model in figure 16. First of all, the area where particles are classified as bcc-like by the neural network is shifted closer to the Gaussians associated to the fcc and hcp regions of the PCA space compared to the classification by the Gaussian Mixture Model. Secondly, the neural network classifies fewer particles as hcp compared to the Gaussian Mixture Model.

We summarize classification by the neural network with the associated confidence levels in the histograms in figure 20. These histograms show the signatures of the four outputs of the neural network during classification of the snapshots from the simulation at 40% supercooling. All simulation snapshots were analysed. For all particles the entire output of the neural network was tracked over the course of the simulation. When a particle is classified into one of the four categories, the highest confidence for that category was stored together with the confidence values for the other three categories. All particles were then divided into bins resulting in the signatures in figure 20.

From the way we labelled our training examples we know, for example, that the best possible match for fcc would have a signature where the confidence for fcc equals one and the confidence for the three other structures would equal zero. The only signature from our

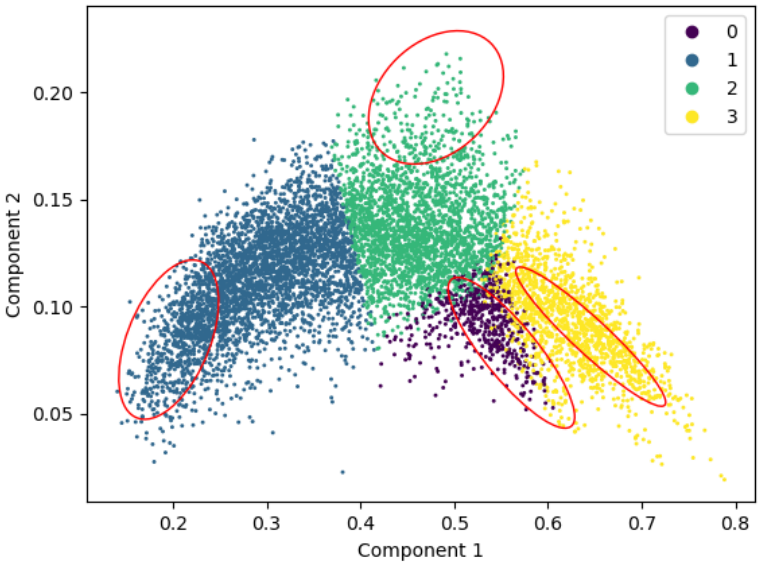
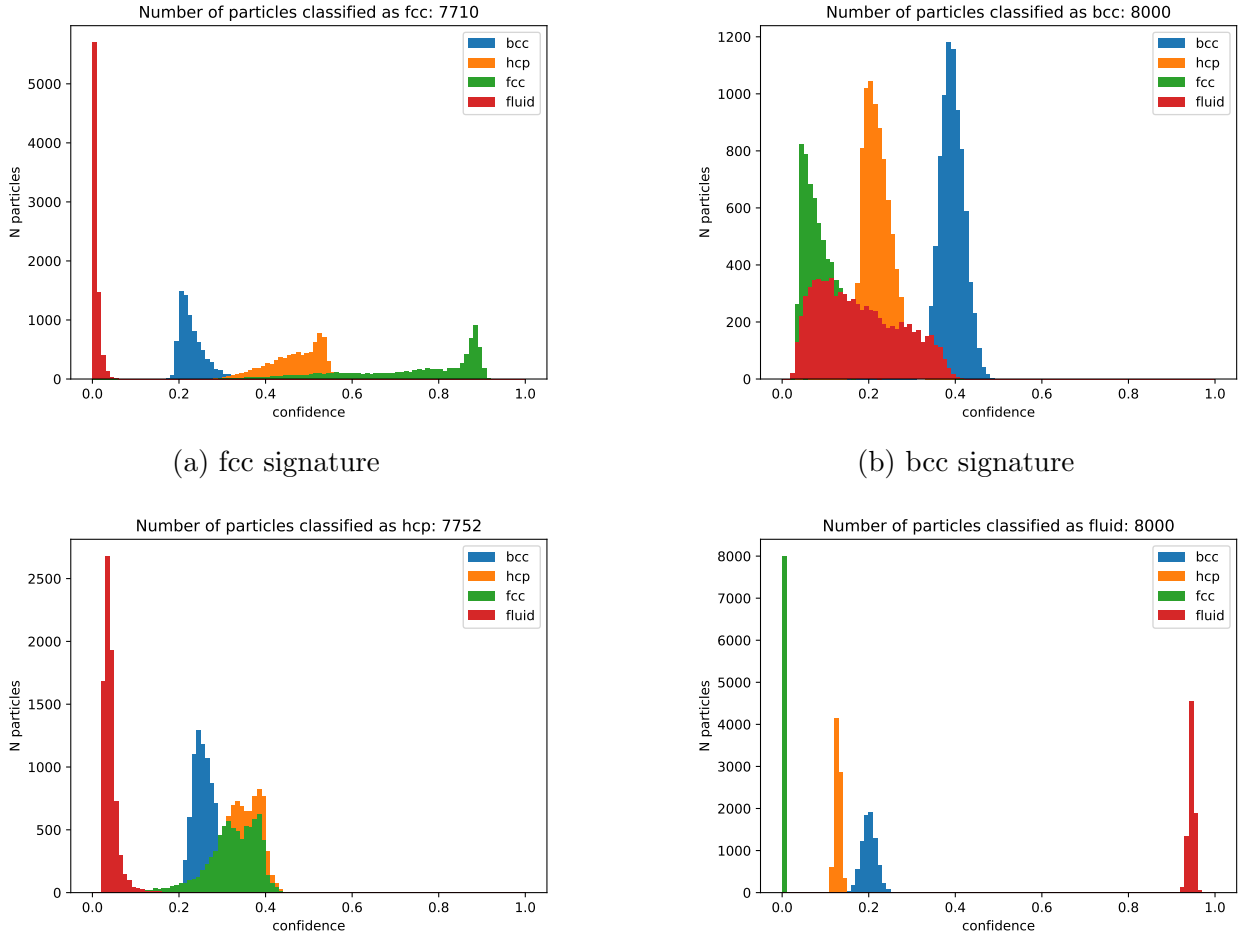


Figure 19: Classification by the neural network plotted on the same PCA space as the classification by the Gaussian Mixture Model. Particles labelled as 0 correspond to hcp, 1 to fluid, 2 to bcc and 3 to fcc. Confidence levels are not represented.

simulation data that somewhat matches that pattern is the signature for fluid-like ordering. The signature for fcc contains a confidence close to one for fcc, but confidence for hcp is also around 0.5. In the hcp signature, the fcc confidence is only slightly lower than the hcp confidence. It seems that our neural network is not able to distinguish hcp over fcc well. This agrees with the results from figure 19.

The signatures in figure 20 reveal several flaws with the use of confidence values to quantify how similar the environment around a particle is to a known reference structure. Firstly, the confidence value with respect to one of the structures does not take the confidence values of the other possible structures into account. For example, from figure 20c it seems that we might not be justified in classifying a particle as hcp-like since the confidence value for fcc structure is similar to the confidence value for hcp structure. Furthermore, the interpretation of confidence values is unclear. The confidence cannot simply be interpreted as a probability and it is therefore uncertain what the minimum confidence should be below which a particle does not belong to that category. Lastly, the behaviour of a neural network in areas where there was no labelled training data available is unpredictable. Without further understanding of the behaviour of the neural network and the interpretation of confidence values, the neural network for classification is not suitable to quantify if crystal nucleation of Lennard-Jones particles happens via bcc.



(a) fcc signature

(b) bcc signature

(c) hcp signature

(d) Fluid signature

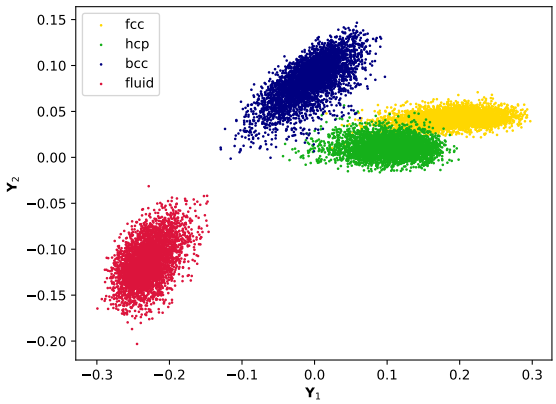
Figure 20: Confidence signatures of classification by neural network of the snapshots generated by the simulation at 40% supercooling.

3.4 Autoencoder Projections

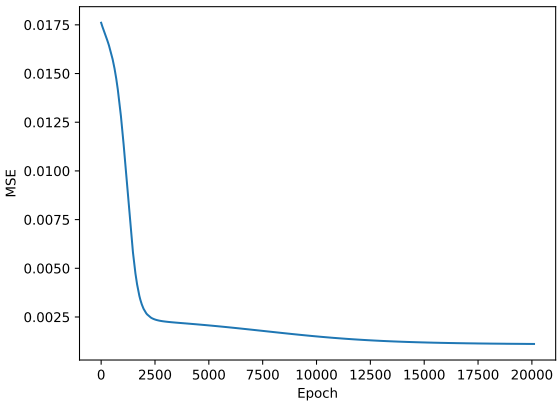
Lastly we compare the two-dimensional spaces found by the autoencoder to the two-dimensional PCA spaces from section 3.2.

The two-dimensional space found by the autoencoder when using $\bar{q}_1$, $\bar{q}_2$, ... , $\bar{q}_{12}$ is shown in figure 21. We can see that the autoencoder is effective in finding a space where the four structure regions are distinguishable from each other. However, the space found by the autoencoder looks similar to the space found by PCA in figure 12. It seems that the non-linearity of the autoencoder does not offer an advantage over PCA.

Use of the non-averaged bond order parameters increases the spatial resolution which might make the non-averaged bond order parameters more appropriate to study possible crystal structures in the interface between the crystal nuclei and the fluid. In section 3.2 we saw that non-averaged bond order parameters are not accurate enough to study crystal structures with PCA. Therefore, we also trained the autoencoder using q_l with $l = 1, 2, ...,12$



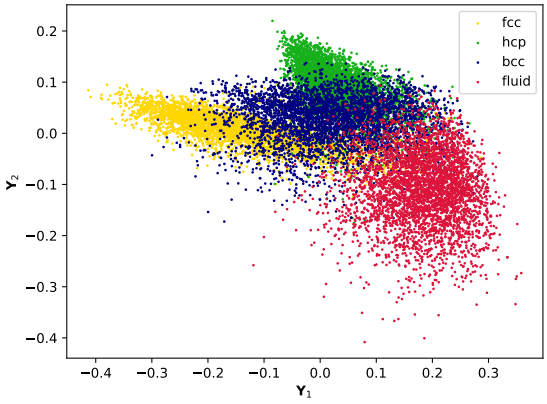
(a) Projections by the encoder.



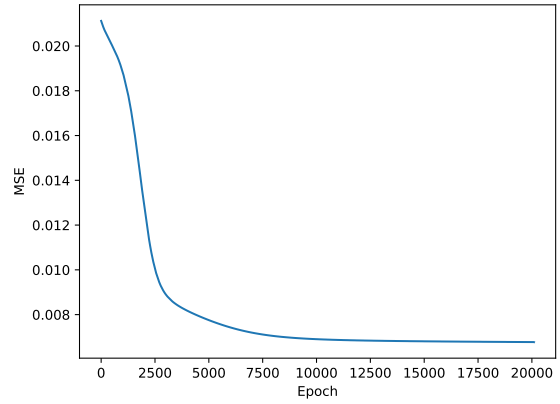
(b) Mean Squared Error over the course of training.

Figure 21: Results from the autoencoder using $\bar{q}_l$. The four known configurations are projected onto the two-dimensional space found by the autoencoder.

to investigate whether the non-linearity of the autoencoder produces a better projection onto two-dimensional space compared to PCA. The projections of q_l onto the two-dimensional space found by the autoencoder are shown in figure 22.



(a) Projections by the encoder.



(b) Mean Squared Error over the course of training.

Figure 22: Results from the autoencoder using q_l. The four known configurations are projected onto the two-dimensional space found by the autoencoder.

The two-dimensional space found by the autoencoder using q_l again looks similar to the PCA space in figure 11. The autoencoder is not able to separate the four structure types either. From the plot of the MSE in figure 22b it seems that we did train the autoencoder long enough. The non-linearity of the autoencoder we used does not offer an advantage over PCA.

4 Conclusion

Understanding the crystallization process of Lennard-Jones particles is important because the Lennard-Jones potential is commonly used to model atomic and molecular interactions. Based on literature, we expected to find that the Lennard-Jones particles crystallize into an fcc crystal. Previous literature also stated that crystal nucleation of Lennard-Jones particles happens via predominantly bcc ordered structures.

We first performed a Principal Component Analysis to distinguish bcc, hcp and fcc ordered structures. The results of the Principal Component Analysis confirmed the formation of an fcc ordered crystal, but we also saw hcp ordered structures. These results also suggested that crystal nucleation does not happen via primarily bcc ordered structures.

Quantifying whether crystal nucleation happens via bcc or not proved to be challenging. The Gaussian Mixture Model clustering method we used was unable to properly quantify how similar to bcc ordered particles were during crystal nucleation. The neural network based classification method we employed also confirmed that Lennard-Jones particles form a primarily fcc ordered crystal, but was not suitable to quantify the degree of bcc ordering during crystal nucleation.

Evaluation of the Gaussian function fitted to the bcc area of the two-dimensional space found by Principal Component Analysis may be useful in quantifying the degree of bcc ordering during crystal nucleation. More investigation needs to be done to assess the validity of using this metric for quantifying bcc ordering. More simulations should also be run at lower supercooling to be more certain that the system is in a metastable state. This can be achieved by biasing the simulation towards crystal nucleus growth.

The difficulties we encountered during our analyses underline the challenge in identifying crystal structures outside the bulk of a crystal. Care must be taken when attempting to identify local crystal structures on a single-particle level.

References

[1] S. Alexander and J. McTague. Should all crystals be bcc? Landau theory of solidification and crystal nucleation. *Physical Review Letters*, 41(10):702–705, 1978.

[2] W. T. Ashurst and W. G. Hoover. Argon shear viscosity via a lennard-jones potential with equilibrium and nonequilibrium molecular dynamics. *Physical Review Letters*, 31(4):206–208, 1973.

[3] Q. Yan and J. J. De Pablo. Hyper-parallel tempering Monte Carlo: Application to the Lennard-Jones fluid and the restricted primitive model. *Journal of Chemical Physics*, 111(21):9509–9516, 1999.

[4] C. Dellago, P. G. Bolhuis, and D. Chandler. Efficient transition path sampling: Application to Lennard-Jones cluster rearrangements. *Journal of Chemical Physics*, 108(22):9236–9245, 1998.

[5] H. Eslami, N. Khanjari, and F. Müller-Plathe. A Local Order Parameter-Based Method for Simulation of Free Energy Barriers in Crystal Nucleation. *Journal of Chemical Theory and Computation*, 13(3):1307–1316, 2017.

[6] P. R. Ten Wolde, M. J. Ruiz-Montero, and D. Frenkel. Simulation of homogeneous crystal nucleation close to coexistence. *Faraday Discuss*, 104:93–110, 1996.

[7] W. Ouyang, B. Sun, Z. Sun, and S. Xu. Entire crystallization process of Lennard-Jones liquids: A large-scale molecular dynamics study. *Journal of Chemical Physics*, 152(5), 2020.

[8] P. J. Steinhardt, D. R. Nelson, and M. Ronchetti. Bond-orientational order in liquids and glasses. *Physical Review B*, 28(2):784–805, 1983.

[9] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[10] D. Frenkel and B. Smit. *Understanding Molecular Simulation; From Algorithms to Applications*. Academic Press, 1996.

[11] M. A. Van Der Hoef. Free energy of the Lennard-Jones solid. *Journal of Chemical Physics*, 113(18):8142–8148, 2000.

[12] J. A. Van Meel, L. Filion, C. Valeriani, and D. Frenkel. A parameter-free, solid-angle based, nearest-neighbor algorithm. *Journal of Chemical Physics*, 136(23), 2012.

[13] W. Lechner and C. Dellago. Accurate determination of crystal structures based on averaged local bond order parameters. *Journal of Chemical Physics*, 129(11), 2008.

[14] N. Guaro. `https://commons.wikimedia.org/wiki/File:GaussianScatterPCA.svg`, 2016. Adapted from work by original author under Creative Commons BY 4.0 license. Accessed on 28-4-2020.

[15] J. Shlens. A Tutorial on Principal Component Analysis. `http://www.cs.cmu.edu/{~}elaw/papers/pca.pdf`, 2005. Accessed on 30-4-2020.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] P. Geiger and C. Dellago. Neural networks for local structure detection in polymorphic systems. *Journal of Chemical Physics*, 139(16), 2013.

[18] E. Boattini, M. Dijkstra, and L. Filion. Unsupervised learning for local structure detection in colloidal systems. *Journal of Chemical Physics*, 151(15):1–12, 2019.

[19] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research*, 9:249–256, 2010.

[20] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. *30th International Conference on Machine Learning, ICML 2013*, (PART 3):2176–2184, 2013.